

EXHIBIT A

Part 3 of 3

US 7,596,784 B2

27

a million and a half simultaneous “views” of an event, there is no way, with prior art systems, to turn those “views” immediately into a million and a half purchases or registrations or any other interaction because a computer system large enough to handle that amount of a load at a single site would be too cost prohibitive. But with the present on-demand method and system 140, the million and a half load is distributed throughout the world wide system of data centers, each housing a plurality edgepoints. Thus, instead of a single server or machine handling a million and a half simultaneous hits, the load is distributed across hundreds of edgepoints and/or servers, which results in thousands or less simultaneous hits per edgepoint 350. A load of tens of thousands of simultaneous hits is manageable for a single server or machine. Thus, the benefits of distributing loads becomes apparent through the scalable, on-demand capacity provided by the present system 140.

A further advantage of the present on-demand method and system is that the user maintains control over their own applications and websites which are deployed over the on-demand system. In prior art systems, the application provider owns their own servers, allowing the application provider with complete control over the application or site. The application provider knows exactly what is being provided. Further, the application provider has direct access to the single server allowing the application provider the ability to monitor the application, the load of the application or site, and the types of interaction occurring with the application or site. However, the prior art systems require the large up-front capital expenditure to initiate and maintain the servers. Further, the prior art systems have either too much capacity and thus wasted capital expenditure, or too little capacity and thus unsatisfied users.

In one embodiment, the present on-demand method and system 140 is designed to allow the application provider with direct control and access to their applications. With the added benefit of being able to monitor specific regions, the application provider has the ability to adjust their applications or websites according to feedback received from a specified region to more accurately address the needs and desires of users in that region by simply adjusting the instances of the appshot housed in edgepoints in those regions of interest. Further, the application provider is able to fully monitor the application. In one embodiment, this is achieved by allowing application providers to create different applications to be deployed geographically as desired. In one embodiment, the on-demand method and system includes: a web-based performance portal to allow the application provider comprehensive statistics on the virtual single server with the additional benefit of obtaining further web response time metrics; alerts based on set bands of acceptable performance; and expense monitoring based on the amount of resources used by the application provider including daily web-based bill review, and alerting of faster-than-expected spending.

Some of the safety precautions or security architecture provided by the novel on-demand method and system 140 are discussed below. The security architecture ensures that edgepoints run registered appshots 220 to prevent hackers from starting other applications; appshots 220 do not allow login, SetUserID, and other such conditions to prevent hackers from breaking out of the appshot control; appshots 220 access limited disk storage, memory, sockets and other such resources to protect user data; the conduit 360 and hub 324 authenticate each other before transfers of data or appshots; appshots 220 are compressed and encrypted when transferred from the conduit 360 to the edgepoints 350; administration is authenticated and changes are audited. The system 140 also

28

prevents denial-of-service attacks because of the size of the distributed on-demand system 140 and the number of edgepoints 350.

In one embodiment, the present on-demand method and system 140 utilizes links from other applications, such as application provider’s home or central web site, to route the user 124 to the desired application 356 stored and maintained on an edgepoint. The system allows an application provider to outsource only a portion of their applications to the on-demand system 140, while still maintaining some of their application processing. For example, an application provider may outsource some of the applications to operate a web site, but, the application provider’s central site is still maintained by the application provider. In an alternative embodiment, an application provider outsources their entire application suite and sites, including their central site, to the on-demand system 140. In one embodiment, the link or pointer from the central site points to a site maintained and controlled by the application provider, but is stored and operated from the resources of the on-demand system 140. When the link or pointer is activated, the on-demand system 140 is accessed and the user 124 is routed to the most optimal edgepoint providing the application desired. In one embodiment, the optimal edgepoint is determined based on network latency and edgepoint load. If the loading on a first edgepoint 350a is too great, the system will route the user 124 to a second edgepoint 350b even though the second edgepoint 350b maybe a further distance way from the user 124 than the first edgepoint 350a. This rerouting is performed because it is worth taking additional latency delays along the routed path to get to the second edgepoint 350b because the second edgepoint 350b is under less load or stress and will provide a superior response, resulting in a superior response even with the added latency delay.

The present on-demand method and system 140 not only provides on-demand, distributed application processing, the on-demand method and system 140 also provides shared resources throughout the distributed on-demand system 140. In one embodiment, because of the unique ability to store an application in a snapshot state, the present invention is also capable of removing an application from resources when the application is not being actively used, thus freeing up the resources. This allows an alternative application to be activated on those resources. Thus providing on-demand, distributed application processing through shared resources which reduces the cost of the resources because a plurality of application providers are utilizing the same resources. In one embodiment, the present invention provides for the ability to return an application not being used into a snapshot state to free up resources for other applications to utilize. Further, in one embodiment, when an active application is returned to a snapshot state freeing up resources, the application provider is no longer charged for the resources that the application was utilizing. The application provider pays for the amount of resources which are actually used by applications distributed by the application provider. The amount of consumed resources are measured in a variety of different ways including: the amount of processor usage; the amount of memory usage; the number of processors operating; the amount of network bandwidth usage; the number of appshots deployed; the density of appshot deployment; and any combination thereof.

Prior art or conventional systems and methods have been developed which distribute content to allow local routing of users. However, these conventional systems and methods do not provide for a method of communicating or returning processed information back to the main site. Prior art out-

US 7,596,784 B2

29

sourced content providers do not provide for processing capabilities of the application providers specific applications. The present invention provides for the separation of applications, the outsourcing of those applications to distribute the load utilizing distributed resources allowing superior performance, without limiting the functions or processing of the applications.

In one embodiment, the present on-demand method and system **140** provides for the scheduling of website or applications to servers and/or resources. The inventive method and system are dynamic and real-time or near-real time. This scheduling of resources is an inversion of the prior-art paradigm of requiring an application to be dedicated to a single server. Typical prior-art systems are configured such that applications are implemented to run on fixed machines or servers. When a request to access an application comes in to a prior art system, the request gets routed to a waiting server. Therefore, the applications on such systems must be active at all times because requests cannot be predicted. Because these applications are fixed or tied to the machine or server, the prior-art server must also be kept running all the time.

The present method and system **140** provides for the dynamic scheduling of a website or application to be processed on demand by restoring an appshot to its running state. Thus, an application can be shut down or removed from server resources until a request for that application is issued. A snapshotted application **220** can be loaded from a shared memory into a server and accompanying resources in less than approximately five seconds and more usually in less than about three seconds, activating what was an idle server. These times are guidelines and not limitations. Thus, the present method and system **140** allows for instant activation of the application on substantially any chosen compute resource. Further, when the application is no longer being used, the present method and system **140** provides the capability to halt the application to free up the resources for another application. Therefore, the system **140** is able to provide economies of scale and favorable pricing to application providers. Attempting to try and achieve this capability through prior art systems is completely impractical because of the amount of time needed to take down an entire application to free up a server and resources along with the amount of time needed to install and activate a new application is completely prohibitive. Thus the present method and system **140** allows for the dynamic scheduling of applications to be processed on demand on optimal compute resources by restoring an appshot to its running state which reverses the paradigm of dedicating servers to applications. Batch processing is therefore well supported

The on-demand network **140** provides the capability of handling more applications per CPU, computer, microprocessor and/or processor than is available through prior art computers, systems or networks by over provisioning the number of applications that are provided by the edgepoint and time multiplexing the use of these applications. In part, this is a result of the "bursting" nature of demand. This is achieved through the unique snapshot/restore ability of the network and edgepoint. Prior art systems cannot provide this capability because of the amount of time needed to take down an application and activate a new application, as well as the loss of data and current state information associated with an application at the time it is taken down. The system **140** provides for the unique ability to quickly halt an application, and store the application and associated states. The halting of the application is achieved without adversely affecting the application or adversely affecting the operation of the application when the application is reactivated or restored for operation. By

30

snapshotting an application, the edgepoint frees up the set of resources for an alternative application. Thus, the edgepoint can multiplex the access and operation of applications without adversely affecting the operation of the application, and without the application and user's knowledge.

In one embodiment, the on-demand method and system **140** is application oriented as apposed to process oriented. The on-demand method and system **140** provides for a virtualization layer in the operating system, such that an application is considered the object of interest, instead of considering the processes as the object of interest. Thus allowing the freezing or halting of an application, and the storage of the application stack, including the different processes, their interprocess communication and its state.

By enabling an application oriented processing network, the method and system enables a higher level of utilization of computing resources. This is a result of increasing the number of applications than can be handled per CPU coupled with the inherently variable nature of demand for computing resources. Prior art systems are characterized by very low average levels of utilization for computing resources. Because it is not possible to quickly take down an application and activate a new application, a single processing resource must necessarily lie idle when demand for the application tied to that resource diminishes.

Application demand varies according to application type, type and geographic location, among other variables. By way of example, demand for enterprise applications usually peaks during business hours whereas demand for consumer-centric web sites may peak during evening hours. Peak demand times will be different in different time zones. Further, demand for processing for certain applications is less time-dependent than others. The present invention enables less time-critical applications (such as batch processing) to be run when demand for more time-critical applications (such as web applications) is low. Because the technology enables the sharing of processors between different applications, this leads to improvements in utilization levels.

FIG. **22** depicts typical exemplary demand situation for two different applications (or customers) across or over a twenty-four hour time period. Demand for Application 1 peaks at **1400** whereas demand for Application 2 peaks at **1900**. With prior art systems, the amount of processing capacity that would be required to satisfy the total demand for both customers or applications is the sum of the total amount of processing capacity that would be required for each of the applications individually (in this case **200** units). With the present invention, the total amount of processing capacity required is equal to the maximum aggregated demand in any time period, in this case **160** units. This is a direct result of the inventive system and method's ability to quickly take down and set up applications resulting from the snapshot/restore technology.

The novel on-demand application processing method and system **140** creates a completely new economic model. The present invention further provides a new technology and method to share compute resources. Still further, the present invention provides the technology and method to bring compute capacity on demand very quickly. The present method and system **140** also provides for dynamic server allocation and resource sharing. Thus, providing on-demand resources at significantly reduced cost to the application provider.

It will be appreciated in light of the description provided herein that the inventive system, method, business model, and operating service moves the provisioning of utility-based or computing utility services to the next level of services, that of a true utility service. Aspects of the invention provide cus-

US 7,596,784 B2

31

tomers the ability to buy just what they need without being trapped into having to pay for capacity they don't use. Essentially computing services are moved from buying fixed capacity, the traditional outsourcing model, to buying variable capacity.

The inventive system and method provides on-demand computing solutions that enable enterprises to improve the server efficiency, application performance and financial return of their information technology (IT) environments. Leveraging an embedded software platform, the inventive system and method enables server infrastructure to be shared securely across applications to offer a range of computing infrastructure management, provisioning and operations solutions to enterprises and service providers with significant application investments. By dynamically allocating, retrieving and tracking computing resources, the inventive system and method enables the first true computing utility. In one embodiment, the system and method provide a service platform offering on-demand web application processing using a utility-based pricing model.

While various aspects of the system and method of the invention have already been described, FIG. 23 provides a diagrammatic illustration showing an exemplary embodiment of a system according to the invention. The diagram illustrates relationships between the Internet (or other network) and routers, distribution nodes (DN), Ethernet Hub, Administrative Nodes (AN), computer nodes (CN) and SAN Hub with associated disk array. Also illustrated are the global dispatcher (GP), deployment center, and NOC. Users and a main site are also coupled to other elements of the system over the Internet or other network connection. An NOC dashboard and a customer dashboard are also provided in this particular system configuration.

The foregoing description of specific embodiments and examples of the invention have been presented for the purpose of illustration and description, and although the invention has been illustrated by certain of the preceding examples, it is not to be construed as being limited thereby. They are not intended to be exhaustive or to limit the invention to the precise forms disclosed, and obviously many modifications, embodiments, and variations are possible in light of the above teaching. It is intended that the scope of the invention encompass the generic area as herein disclosed, and by the claims appended hereto and their equivalents.

Having disclosed exemplary embodiments and the best mode, modifications and variations may be made to the disclosed embodiments while remaining within the scope of the present invention as defined by the following claims.

What is claimed is:

1. A method for providing distributed, on-demand application processing, comprising:

allowing a first application provider to deploy at least a first application onto a network, wherein the network includes distributed compute resources configured to provide application execution;

receiving a request for execution of the first application from the first application provider; and utilizing the distributed compute resources for execution of the first application in response to receiving the request for execution of the first application from the first application provider;

metering and monitoring an amount of the compute resources utilized in execution of the first application; charging the first application provider based on the amount of compute resources utilized in execution of the first application;

32

increasing the amount of compute resources utilized in execution of the first application by a first set of compute resources; and increasing the amount charged to the first application provider based on the first set of compute resources.

2. The method as claimed in claim 1, further comprising: allowing the first application provider to update the first application.

3. The method as claimed in claim 2, further comprising: allowing the first application provider to monitor the first application.

4. The method as claimed in claim 3, further comprising: allowing the first application provider to replace the first application with a second application.

5. The method as claimed in claim 1, further comprising: allowing an entity to access the compute resources over the network to interact with the first application.

6. The method as claimed in claim 1, wherein increasing the amount of computer resources includes increasing the amount of compute resources due to an increased demand for the first application.

7. The method as claimed in claim 1, further comprising: reducing the amount of compute resources utilized in execution of the first application by a second set of compute resources; and reducing the amount charged to the first application provider based on the second set of compute resources.

8. The method as claimed in claim 1, further comprising: providing a second application provider with access to the network;

allowing the second application provider to distribute at least a third application onto the network;

allowing an entity to access the compute resources over the network to interact with the third application;

monitoring the amount of the compute resources utilized for providing application processing of the third application; and

charging the second application provider based on the amount of compute resources utilized in processing the third application.

9. A method of providing on-demand computational resources over a distributed network, the method comprising: providing an application provider with access to a distributed network;

through the network the application provider dictating at least a first portion of the distributed network to receive at least one application; and

distributing the application onto computational resources within the first portion of the distributed network dictated to receive the application.

10. The method as claimed in claim 9, further comprising: the application provider dictating a limit on an amount of the computational resources to be utilized in providing at least one entity with access to the application for processing of the application.

11. The method as claimed in claim 10, further comprising: through the network, the application provider updating the first application.

12. The method as claimed in claim 10, further comprising: metering and monitoring the amount of computational resources utilized in providing the at least one entity with access to application; and

charging the application provider based on the amount of computational resources utilized in providing the at least one entity with access to the application.

13. The method as claimed in claim 12, further comprising: metering and monitoring demand for the application; and

US 7,596,784 B2

33

adjusting the amount of computational resources utilized in providing the at least one entity with access to the application based on the demand.

14. The method as claimed in claim 13, wherein adjusting the amount of computational resources utilized comprises not exceeding the limit of the computational resources dictated by the application provider.

15. The method as claimed in claim 13, wherein adjusting the amount of computational resources utilized further comprises:

initiating an additional instance of the application if the demand for the application exceeds a first threshold; and halting an instance of the application if the demand for the application falls below a second threshold.

16. The method as claimed in claim 15, wherein initiating an instance of the application comprises restoring a snapshotted application; and wherein halting an instance of the application comprises snapshotting an active application.

17. The method as claimed in claim 10, further comprising: the application provider monitoring the amount of computational resources utilized in providing the at least one entity with access to the application; and the application provider adjusting the limit on the amount of computational resources available to provide the at least one entity with access to the application.

18. The method as claimed in claim 17, wherein: the first threshold is based on response time of the application.

19. An apparatus for providing on-demand compute resources, comprising:

a plurality of compute resources including at least one memory, wherein the plurality of compute resources are distributed across a network, wherein the plurality of compute resources are coupled to allow communication between at least first compute resources and second compute resources; a conduit coupled to the network, and

configured to provide an application provider access to the network to distribute at least a first application onto the network and request execution of the first application using the plurality of compute resources;

a first resource manager coupled with at least the first compute resources, wherein the first resource manager is configured to activate at least a first set of the first compute resources for processing of the first application when demand for the first application exceeds a first threshold;

and a metering module coupled with the first compute resources, wherein the metering module is configured to monitor the first compute resources including the first set of the first computer resource being activated for processing of the first application, and wherein the metering module is configured to determine an amount to bill the first application provider based on the first set of the first compute resources utilized;

wherein the first resource manager is configured to activate a second set of the first compute resources for processing

34

of the first application when demand for the first application exceeds a second threshold;

and wherein the metering module is configured to monitor the second set of the first computer resource being activated for processing of the first application, and wherein the metering module is configured to determine an increased amount to bill the first application provider based on the second set of the first compute resources utilized.

20. The apparatus as claimed in claim 19, wherein: the conduit is further configured to provide the application provider with access to the network to update the first application.

21. The apparatus as claimed in claim 20, wherein: the conduit is further configured to provide the application provider with access to the network to replace the first application with a second application.

22. The apparatus as claimed in claim 20, wherein: the first compute resources includes at least a first and second set of compute resources, where the first and second set of the first compute resources are configured to be activated and deactivated based on a demand and to provide application processing for at least the first application.

23. The apparatus as claimed in claim 22, further comprising wherein:

the first resource manager coupled with the first compute resources, and is configured to monitor demand for at least the first application such that the resource manager activates and deactivates at least one of the first and second sets of the first computer resources based on demand.

24. The apparatus as claimed in claim 23, wherein the first resource manager is further configured to monitor the amount of the first compute resources utilized in processing the first application; and the apparatus further comprising:

wherein the metering module coupled with the first resource manager, and is configured to receive the amount of the first compute resources utilized and to determine an amount to bill the first application provider based on the amount of the first compute resources utilized.

25. The apparatus as claimed in claim 19, wherein: the first resource manager is configured to deactivate one of the first and second sets of the first compute resources processing the first application when the demand for the first application falls below a third threshold; and

the metering module registers a decreased amount of compute resources utilized for application processing of the first application based on the deactivation of one of the first and second sets of the first computer resources being deactivated, and the metering module is configured to determine a reduce amount to billed the first application provider based on the decreased amount of the first compute resources utilized.

* * * * *

UNITED STATES PATENT AND TRADEMARK OFFICE

CERTIFICATE OF CORRECTION

PATENT NO. : 7,596,784 B2
APPLICATION NO. : 09/950559
DATED : September 29, 2009
INVENTOR(S) : Abrams et al.

Page 1 of 1

It is certified that error appears in the above-identified patent and that said Letters Patent is hereby corrected as shown below:

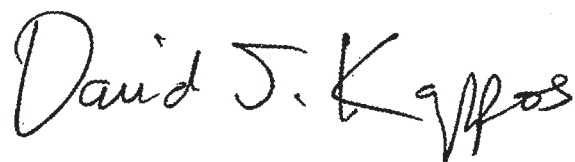
On the Title Page:

The first or sole Notice should read --

Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b)
by 2232 days.

Signed and Sealed this

Twenty-eighth Day of September, 2010

A handwritten signature in black ink that reads "David J. Kappos". The signature is written in a cursive, flowing style with a large initial 'D' and a stylized 'K'.

David J. Kappos
Director of the United States Patent and Trademark Office